



Integrating xiRAID Classic 4.3.1 Into a Pacemaker Cluster

Contents

Integrating xiRAID Classic 4.3.1 Into a Pacemaker Cluster.....	3
Introduction.....	3
Prerequisites.....	3
xiRAID Classic Installation (HA).....	4
Updating xiRAID Classic 4.3.0 to xiRAID Classic 4.3.1 (HA).....	5
Csync ² Installation & Configuration.....	6
Installing Csync ²	6
Csync ² Configuration.....	8
Configuring Scheduled Synchronization.....	9
Troubleshooting Synchronization Errors.....	10
Managing RAIDs in a Cluster.....	11
RAID and Cluster Resource Creation.....	11
Automatic Drive Replacement in a Cluster.....	12
Changing RAID parameters.....	13
Deleting RAIDs.....	14
Additional Network Configuration.....	14
Generating SSL Certificates for Cluster Nodes.....	14
Configuring Authentication Settings.....	19

Integrating xiRAID Classic 4.3.1 Into a Pacemaker Cluster

Instructions on how to integrate xiRAID Classic into an existing Pacemaker cluster.

Introduction

xiRAID Classic can operate within a High Availability (HA) cluster, ensuring data integrity and availability for users by distributing data and system components across two nodes. To set up a High Availability cluster for xiRAID Classic, configure and connect two cluster nodes to a shared set of drives. In case of a node failure, the xiRAID Classic components can migrate to other operational nodes within the cluster, maintaining continuous operations without disruption.

The management of xiRAID Classic components within an HA cluster is conducted by the Pacemaker cluster manager, which coordinates failover mechanisms and ensures transitions of RAIDs between nodes. File synchronization among clustered nodes can be facilitated by the Csync² utility, which includes conflict detection, file deletion detection, and update handling. However, users have the option to choose any other utility for this purpose.

This guide provides instructions on integrating xiRAID into an existing Pacemaker cluster.

Prerequisites

The following prerequisites must be fulfilled to successfully integrate xiRAID Classic 4.3.1 into an existing two-node Pacemaker cluster:

1. The Pacemaker cluster software is installed and configured to run on two nodes.
2. Fencing (STONITH) is configured in the cluster.
3. Nodes in the cluster meet the requirements outlined in the xiRAID Classic Software Requirements document.
4. The Pacemaker cluster manager version is 2.1.6 or later.
5. The firewall on both nodes is configured in a way that allows nodes to communicate with each other.
6. The time is synchronized across all the cluster nodes.
7. Disks that will be used to create xiRAID RAID(s) are accessible to both nodes in the cluster.
8. The jq package (command-line JSON processor) version 1.6 or later is installed on both nodes. Execute the following command to install the jq package:

- a. RHEL, RHEL-based systems, and Oracle Linux:

```
# dnf install jq
```

- b. Ubuntu and Proxmox:

```
# apt install jq
```



It is crucial that all of the prerequisites above are fulfilled before you proceed to the next section.

xiRAID Classic Installation (HA)

1. Follow the instructions in the xiRAID Classic Installation Guide to install xiRAID Classic 4.3.1 on both nodes in the cluster.
2. Follow the instructions in the xiRAID Classic Administrator's Guide to apply a xiRAID license to both nodes in the cluster.



Each node in the cluster requires a separate license.

3. On both nodes, install the xiRAID resource agent scripts for Pacemaker.

During the xiRAID installation, these scripts was added to the `/etc/xraid/agents` directory. For this agent to work with Pacemaker, you need to create a symbolic link from `/etc/xraid/agents` to the directory with Pacemaker resource agent scripts `/usr/lib/ocf/resource.d`:

```
# ln -s /etc/xraid/agents /usr/lib/ocf/resource.d/xraid
```

Updating xiRAID Classic 4.3.0 to xiRAID Classic 4.3.1 (HA)

1. Place one of the nodes into standby mode:

```
pcs node standby <node1-name>
```

2. Wait for all RAID resources to move to the second node.

3. Stop the Pacemaker services on the node:

```
pcs cluster stop <node1-name>
```

4. Follow the instructions in the xiRAID Classic 4.3.0 to xiRAID Classic 4.3.1 Update Guide to update the first node to the new version of xiRAID Classic.

5. Restart the Pacemaker services on the node:

```
pcs cluster start <node1-name>
```

6. Remove the first node from standby mode:

```
pcs node unstandby <node1-name>
```

7. Wait for RAID resources to move to the first node and confirm that they are fully operational.

8. Place the second node into standby mode:

```
pcs node standby <node2-name>
```

9. Wait for all RAID resources to move to the first node.

10. Stop the Pacemaker services on the node:

```
pcs cluster stop <node2-name>
```

11. Follow the instructions in the xiRAID Classic 4.3.0 to xiRAID Classic 4.3.1 Update Guide to update the second node to the new version of xiRAID Classic.
12. Restart the Pacemaker services on the node:

```
pcs cluster start <node2-name>
```

13. Remove the second node from standby mode:


```
pcs node unstandby <node2-name>
```

Csync² Installation & Configuration

Csync² is a utility for asynchronous file synchronization. Csync² is used to synchronize RAID config files between nodes in a cluster.

Installing Csync²

This section describes how to install Csync².

 Csync² must be installed on both nodes in the cluster.

RHEL-Based Systems

On RHEL-based systems, Csync² is not available from the official repositories, but it is included in the xiRAID repository. The xiRAID Classic repository is installed during the xiRAID Classic installation, so no additional steps are required to install Csync². Simply run the following command:

```
# dnf install csync2
```

Oracle Linux

On Oracle Linux, Csync² is not available from the official repositories, but it is included in the xiRAID repository. The xiRAID Classic repository is installed during

the xiRAID Classic installation, so no additional steps are required to install Csync². Simply run the following command:

```
# dnf install csync2
```

Ubuntu

On Ubuntu, Csync² can be installed from the official repository:

```
# apt install csync2
```

Instructions in the following sections assume that you've installed Csync² from the source code or the xiRAID repository. If you've installed Csync² from the official repository, note the following differences:



- The configuration file for Csync² should not be created in `/usr/local/etc/` as described in the [Csync² Configuration](#) section. Instead, refer to [the official documentation](#) for the correct location of the configuration file.
- The path to the Csync² executable is `/usr/sbin/csync2` instead of `/usr/local/sbin/csync2`.

Installation From Source

1. For RHEL, RHEL-based systems, and Oracle Linux, install the required dependencies:

```
# dnf install automake byacc flex git gcc gnutls-devel librsync  
librsync-devel libsqlite3x-devel libpq-devel make mysql-devel
```

2. Clone the Csync² repository:

```
# git clone https://github.com/LINBIT/csync2.git
```

3. Compile and install Csync²:

```
# cd csync2
# ./autogen.sh
# ./configure
# make
# make install
```

4. Create the systemd service unit configuration file:

```
# vi /etc/systemd/system/csync2.service
```

5. Paste the following content into the file:

```
[Unit]
Description=csync2 file synchronization tool using librsync and
current state databases
After=network.target

[Service]
Type=idle
ExecStart=/usr/local/sbin/csync2 -ii -l
Restart=on-failure

[Install]
WantedBy=pcsd.service
```

6. Start the service:

```
# systemctl start csync2.service
```

7. Configure the service to be started automatically at boot:

```
# systemctl enable csync2.service
```

Csync² Configuration



Steps 1-4 should only be performed on the first node.

1. Create the configuration file:

```
# vi /usr/local/etc/csync2.cfg
```

2. Paste the following content into the file:

```
noss1 * *;  
group csxiha {  
  host <node1_hostname>;  
  host <node2_hostname>;  
  key /usr/local/etc/csync2.key_ha;  
  include /etc/xiraid/pools;  
  include /etc/xiraid/raids;  
  exclude /etc/xiraid/pools/*.bak;  
  exclude /etc/xiraid/raids/*.bak;  
}
```

3. Generate the encryption key:

```
# csync2 -k /usr/local/etc/csync2.key_ha
```

4. Copy the configuration file and the generated key to the second node:

```
# scp /usr/local/etc/csync2.cfg /usr/local/etc/csync2.key_ha  
<node2_hostname>:/usr/local/etc/
```

5. Configure the firewall on both nodes to allow incoming TCP traffic on port 30865.

Firewall-cmd example:

```
# firewall-cmd --add-port=30865/tcp  
# firewall-cmd --permanent --add-port=30865/tcp
```

Configuring Scheduled Synchronization

Configure the synchronization task to run every minute using cron:

1. Open the crontab file:

```
# crontab -e
```

2. Add the following line to the file to schedule the synchronization task to run every minute:

```
* * * * * /usr/local/sbin/csync2 -x
```



If necessary, update the path to csync2 in the cron job.

3. Save the file.

Create a script to immediately sync the configuration files when any configuration file is updated:

1. Create the script file:

```
# vi /etc/xiraid/config_update_handler.sh
```

2. Paste the following content into the file:

```
#!/usr/bin/bash
/usr/local/sbin/csync2 -xv
```



If necessary, update the path to csync2 in the script.

3. Make the script executable:

```
# chmod +x /etc/xiraid/config_update_handler.sh
```

Troubleshooting Synchronization Errors

It is recommended to periodically check for configuration synchronization errors. If any errors are detected, contact the support team to prevent a scenario where failover might become impossible.

The following error message in the logs indicates synchronization issues:

```
Config update handler is failed
```

To verify that there are no synchronization errors, manually run the synchronization command and check the output:

```
# csync2 -xv
```

You should see the following if there are no errors:

```
Connection closed.  
Finished with 0 errors.
```

Managing RAIDs in a Cluster

In this chapter you will learn how to create, manage and delete xiRAID Classic RAID objects in HA cluster configuration.

RAID and Cluster Resource Creation

Follow the steps below to create a xiRAID RAID and add a Pacemaker resource to manage it:

1. Disable RAID autostart:

```
# xicli settings cluster modify --raid_autostart 0  
# xicli settings cluster modify --pool_autoactivate 0  
# systemctl restart xiraid-server.service
```



It is important to disable xiRAID autostart before creating any RAIDs to prevent data corruption due to accidental active-active mode.

2. Create a xiRAID RAID by following the instructions in the xiRAID Classic Administrator's Guide.
3. Get the RAID UUID:

```
# xicli raid show -n <raid_name> -f json
```

4. Unload the RAID:

```
# xicli raid unload -n <raid_name>
```

5. Add the Pacemaker resource to manage the RAID using the RAID name (<raid_name>) and UUID (<raid_uuid>):

```
# pcs resource create <raid_name> ocf:xraid:raid \  
    name=<raid_name> uuid=<raid_uuid> \  
    op monitor interval=5s \  
    meta migration-threshold=3
```



You can set the *migration-threshold* value for a resource, allowing it to move to a new node after a specified number of failures. When working with xiRAID RAIDs, it is recommended to set this parameter to 3. To learn more, see [Moving Resources Due to Failure](#).

Automatic Drive Replacement in a Cluster

In a high availability configuration, administrators can create a spare pool of drives that is accessible by any node within the cluster. Although spare pools are created on one node, they are available to all nodes in the cluster. While a spare pool is technically active on only one node at any given time, if a drive fails on a node, it can be replaced using a drive from the spare pool currently active on another node.

Network Configuration

For a node to use a drive from a spare pool hosted on a different node, cluster nodes must communicate through the gRPC API. Additional network configuration is required to enable secure communication between nodes in the cluster. For detailed network configuration instructions, refer to [Additional Network Configuration](#).

Creating a Spare Pool



To ensure automatic drive replacement works, please first complete the steps outlined in the [Additional Network Configuration](#) section.

First, run the following command **on both nodes** to set the `pool_autoactivate` parameter to 0:

```
# xicli settings cluster modify --pool_autoactivate 0  
# systemctl restart xiraid-server.service
```

Then follow the steps below to create a spare pool to use it in a cluster. These steps can be performed on any node within the cluster.

1. Create a spare pool using the `xicli pool create` command:

```
xicli pool create -n <spare_pool_name> -d <list_of_drives>
```

2. Create a Pacemaker resource to manage this spare pool:

```
pcs resource create <spare_pool_name> ocf:xraid:pool \  
    name=<spare_pool_name> \  
    op monitor interval=5s \  
    meta migration-threshold=5
```

3. Verify the resource status by running the following command:

```
pcs status
```

4. Use the `xicli pool show` command to ensure that the spare pool is active only on the node where the pool resource is started, and not on any other nodes:

```
xicli pool show
```

5. Use the `xicli raid modify` command to assign this spare pool to a RAID:

```
xicli raid modify -n <raid_name> -sp <spare_pool_name>
```

Changing RAID parameters



Please ensure that you change the RAID parameters on the cluster node where it is currently active.

Change the existing RAID's parameters using the instructions provided in the xiRAID Classic Administrator's Guide.

Currently, changing the RAID name is not supported. To change the RAID name, you need to delete the Pacemaker resource that manages the RAID, update the RAID name, and then create a new Pacemaker resource with the new name.

Deleting RAIDs

To delete a RAID in HA cluster:

1. Delete the Pacemaker resource that manages the RAID:

```
# pcs resource delete <raid_name>
```

2. Delete the RAID using the instructions provided in the xiRAID Classic Administrator's Guide. Example:

```
# xicli raid destroy -n <raid_name>
```

Additional Network Configuration

In some scenarios, nodes in a cluster need to communicate over the gRPC API. To enable secure communication, it is necessary to issue SSL certificates and configure authentication settings.

Currently, this additional network configuration is required only for enabling [Automatic Drive Replacement in a Cluster](#). It allows nodes to communicate changes to the spare pool, ensuring that all nodes in the cluster are aware of its current state and the availability of drives.

Generating SSL Certificates for Cluster Nodes

Steps outlined in this section can be performed on one of the cluster nodes or another machine. Some of the generated files will need to be transferred to all nodes, as described in [Transferring Files to Cluster Nodes](#).

Creating a Self-Signed Root Certificate Authority Certificate

1. Create a directory for certificates:

```
# mkdir certs && cd certs
```

2. Create a self-signed root Certificate Authority certificate:

```
# openssl req -newkey rsa:2048 -nodes -keyout ca_key.pem -x509  
-days 3654 -out ca-cert.crt
```

Creating Configuration Files for Certificate Details

Create a configuration file with certificate details for each node.

Below are two examples files for node1 and node2 with IP addresses `172.168.1.1` and `172.168.1.2` respectively. Make sure to update the `[req_distinguished_name]` and `[alt_names]` section as necessary.

node1.conf

```
[req]
distinguished_name = req_distinguished_name
req_extensions = req_ext
prompt = no

[req_distinguished_name]
C   = Country
ST  = State
L   = Locality
O   = Organization
OU  = Organizational Unit
CN  = localhost

[req_ext]
subjectAltName = @alt_names

[alt_names]
DNS.1 = localhost
DNS.2 = node1
DNS.3 = 172.16.1.1
```

node2.conf

```
[req]
distinguished_name = req_distinguished_name
req_extensions = req_ext
prompt = no

[req_distinguished_name]
C   = Country
ST  = State
L   = Locality
O   = Organization
OU  = Organizational Unit
CN  = localhost

[req_ext]
subjectAltName = @alt_names

[alt_names]
DNS.1 = localhost
DNS.2 = node2
DNS.3 = 172.16.1.2
```

Generating Private Keys and Certificate Signing Requests for Cluster Nodes

1. Generate a private key for each node. For our example with two nodes:

```
# openssl genrsa -out node1-server-key.key 4096
# openssl genrsa -out node2-server-key.key 4096
```

2. Generate a certificate signing request (CSR) for each private key:

```
# openssl req -new -sha256 \
    -out node1-server-csr.csr \
    -key node1-server-key.key \
    -config node1.conf
# openssl req -new -sha256 \
    -out node2-server-csr.csr \
    -key node2-server-key.key \
    -config node2.conf
```

Signing Cluster Node Certificate Signing Requests

Sign the CSRs created in the previous step with the root CA certificate:

```
# openssl x509 -req -in node1-server-csr.csr \
    -CA ca-cert.crt \
    -CAkey ca_key.pem \
    -CAcreateserial \
    -out node1-server-cert.crt \
    -days 365 -extfile node1.conf -extensions req_ext
# openssl x509 -req -in node2-server-csr.csr \
    -CA ca-cert.crt \
    -CAkey ca_key.pem \
    -CAcreateserial \
    -out node2-server-cert.crt \
    -days 365 -extfile node2.conf -extensions req_ext
```

Transferring Files to Cluster Nodes

1. Copy `ca-cert.crt` to the `/etc/xraid/crt` directory on all cluster nodes.
2. Copy the `<node-name>-server-key.key` and `<node-name>-server-cert.crt` files to the `/etc/xraid/crt` directory on the respective nodes. For example, for node1, you would copy `node1-server-key.key` and `node1-server-cert.crt` to `/etc/xraid/crt` on node1.
3. Rename the `<node-name>-server-key.key` and `<node-name>-server-cert.crt` files after copying them to their respective nodes. Remove `<node-name>` from the file names, so you end up with `server-key.key` and `server-cert.crt`.

Configuring Authentication Settings

Perform the following step on each cluster node:

```
xicli settings auth modify --host <node IP address> --port <port>
```

`<node IP address>` is the IP address that other nodes will use to connect to this node.

For example:

```
xicli settings auth modify --host 172.16.1.1 --port 6066
```



Ensure that the firewall is configured to allow communication between nodes over the specified port.